

Tema 1

1	2	3	4	Calificación
B B	B	B B	B B	9 (nueve)

APELLIDO Y NOMBRE: Sebastián Ariel Sepeiling
Nº DE LIBRETA: 703/13 HOJAS: 4

Lógica y Computabilidad – Recuperatorio del Segundo Parcial
1er Cuatrimestre de 2018 – 20/7/2018

Empezar cada ejercicio en hoja nueva. Numerar y poner nombre a todas las hojas. Se pueden utilizar todas las macros y los resultados sobre funciones computables y recursivas primitivas vistos en clase. Justificar todas las respuestas.

1. Sea ψ una función unaria recursiva primitiva. Definimos la función binaria f como

$$f(n, x) = \psi^n(x) + x^n,$$

es decir, $f(0, x) = x + 1$, $f(1, x) = \psi(x) + x$, $f(2, x) = \psi(\psi(x)) + x^2$, etc.

- Probar que f es computable.
- Probar que f es recursiva primitiva.

2. Sea F la función binaria dada por

$$\begin{aligned} F(0, 0) &= 0, \\ F(0, 1) &= 1, \\ F(0, y) &= F(0, y-1) + F(0, y-2), \quad y > 1, \\ F(x, y) &= F(x-1, F(x-1, y+1)), \quad x > 0. \end{aligned}$$

¿Es F una función computable?

3. Sea B un conjunto recursivo y P el predicado dado por $P(x) \leftrightarrow x \in B$. Definamos

$$g(x) = \text{HALT}(x, P(x)) \quad \text{y} \quad h(x) = \Phi(P(x), x).$$

- Probar que g es computable.
- Probar que h no es computable.
- ¿Es h parcialmente computable?

4. Sea P un predicado unario computable y f la función definida como

$$f(n) = \begin{cases} 4 \cdot 3^{29} & \text{si } P(n) \\ 9 \cdot 2^{29} & \text{si } \neg P(n). \end{cases}$$

Definimos además $g(n) = \Phi(f(n), f(n))$.

- Probar que g es parcialmente computable.
- ¿Es $\text{Dom}(g)$ un conjunto recursivo?

1

a) f es computable por el siguiente programa:

(Se asumen macros para

$$z_1 \leftarrow x_1$$

Asumiendo que todas
las variables locales
se inician en 0,
sobre el z_2 .

- asignación

- salto incondicional

- suma

- multiplicación

- ψ (es recursión primitiva, y por
ende computable) ✓

$z_2 \leftarrow 1$ $z_2 \leftarrow z_2 + 1$

$$y \leftarrow x_2$$

[A] IF $z_1 \neq 0$ GOTO B

GOTO C

[B] $y \leftarrow \psi(y)$

$$z_2 \leftarrow z_2 \times x_2$$

$$z_1 \leftarrow z_1 - 1$$

GOTO A

[C] $y \leftarrow y + z_2$ ✓

Todos los macros usados son totales. por ende, este programa
es total. ✓

b) Podemos definir f por recursión

$$f(0, x) = h(x) = x + 1$$

$$f(y+1, x) = g(f(y, x), y, x) = \psi(f(y, x) - x^y) + x^{y+1}$$

por inducción, el caso base es trivial, y

$$f(y+1, x) = \psi(f(y, x) - x^y) + x^{y+1} \text{ por}$$

HI
~~= $\psi(x^y)$~~

$$= \psi(\psi^y(x) + x^y - x^y) + x^{y+1}$$

$$= \psi(\psi^y(x)) + x^{y+1} = \psi^{y+1}(x) + x^{y+1} \quad \checkmark$$

por ende, al ser composición de funciones recursivas primitivas

(ψ , suma y potenciación) f también es recursiva primitiva \square

En particular, es computable y se deduce el ítem anterior.

2) Podemos reducir F usando la siguiente función

$$F(0, y) = \text{fib}(y)$$

Donde fib (la función de Fibonacci), se define

$$\text{fib}(0) = 0 \quad \text{fib}(1) = 1 \quad \text{fib}(x+2) = \text{fib}(x+1) + \text{fib}(x) \quad \checkmark$$

se sabe que $\text{fib}(x)$ es computable $\forall x \in \mathbb{N}$ ~~(por el teorema)~~ ✓

Luego, tenemos

$$F(0, y) = \text{fib}(y)$$

$$F(x+1, y) = F(x, F(x, y+1)) \quad \checkmark$$

Aplicando recursión sobre x

$$h(y) = \text{fib}(y)$$

$$g(x, y, F(x, y)) = F(x, F(x, y+1))$$

veamos que F es composición de funciones computables totales

(suma y Fibonacci)

Luego, F es computable. ✓

3) a) Sabemos que Halt se define

$$\text{HALT}(x, y) = \begin{cases} 1 & \text{si } y = \#(P) \text{ y } \psi_P^{(y)}(x) \downarrow \\ 0 & \text{si } y \neq \#(P) \text{ y } \psi_P^{(y)}(x) \uparrow \end{cases}$$

En este caso, $g(x) = \text{HALT}(x, P(x))$. Es decir que

$$y = \#(P) = P(x)$$

Por lo tanto, el programa solo puede tomar 2 valores

~~con~~ (omitendo el uso de la instrucción $y \leftarrow y$ o $\#(I) = 0$)

$\#(P) = 0$ $\#(P) = 1$	}	$\#(P) = 1$ $\#(P) = 0$
$y \leftarrow y$ ✓		$y \leftarrow y + 1$ ✓

Es decir, $g(x) = \begin{cases} \text{HALT}(x, 1) & \text{si } P(x) \\ \text{HALT}(x, 0) & \text{cc.} \end{cases} = 1 \quad \forall x \in \mathbb{N}$

Por ende, $g(x)$ es computable \square ✓

b) $h(x)$, por otro lado, está definido como $\Phi(P(x), x) = \psi_x^{(x)}(P(x))$

Si tomamos el programa P

$y = y + 1$
[A] IF $y \neq 0$ GOTO A

El mismo no es computable. Como $x \in \mathbb{N}$, ~~Φ~~ si $x = \#(P)$, $h(x) \uparrow$. Ergo, $h(x)$ no es computable ✓

→ Todos los programas computan a alguna función (total o parcial)
En este caso computa a la función que no está definida en ningún lado.

c) Supongamos que h es parcialmente computable, en dominio.

$$\text{Dom}(h) = \{x \in \mathbb{N} / \Phi(P(x), x) \downarrow\}$$

Esto es equivalente a

$$\text{Dom}(h) = \{x \in \mathbb{N} / \cancel{\Phi(P(x), x) \downarrow} \vee (\Phi(P(x), x) \downarrow \wedge \Phi(x, x) \downarrow)\}$$

$$(P(x) \wedge \Phi_x(x) \downarrow) \vee (\neg P(x) \wedge \Phi_x(x) \downarrow)\}$$

~~Esto es equivalente a~~

$$\text{Dom}(h) = \cancel{\{x \in \mathbb{N} / \Phi(P(x), x) \downarrow\}}$$

$$\{x \in B : \Phi \in \text{Dom } \Phi_x\} \cup \{x \in B^c : 0 \in \text{Dom } \Phi_x\}$$

Esos conjuntos son recursivos enumerables, por lo que $\text{Dom}(h)$ también lo es.

¿Por qué?

NO.

por ende, h es parcialmente computable

Una función puede ser o no parcialmente computable aunque su dominio sea r.e.

Hay que ver si hay un programa que compute parcialmente a h .

4a) $g(n)$ puede resultar en la ejecución de dos programas distintos:

si $P(n)$

$Y \leftarrow Y + 1$

[A] IF $Y \neq 0$ GOTO A

Este programa no termina porque al llegar a [A], $Y = 1$.

si $\neg P(n)$

[A] IF $Y \neq 0$ GOTO A

$Y \leftarrow Y + 1$

Este programa finaliza porque $Y = 0$ al comenzar el programa

Esto resulta de la codificación de las instrucciones #2 y #29

Por lo tanto:

$$g(n) = \begin{cases} \uparrow & \text{si } P(n) \\ 1 & \text{si } \neg P(n) \end{cases}$$

Es decir, $g(n)$ es parcialmente computable

b) $\text{Dom}(g)$ está definido por P el predicado de la siguiente forma

$$\text{Dom}(g) = \{x \in \mathbb{N} / \neg P(x)\}$$

Dado que P es un predicado computable, tanto $\text{Dom}(g)$ como su complemento son recursivos enumerables.

Es decir, $\text{Dom}(g)$ es recursivo \square